University of Khartoum

Faculty of Engineering and Architecture

Department of Electrical and Electronics Engineering

M.Sc in Computer Architecture and Networks

Testing Artificial Intelligent Tools for LAN Design

By:

Mohamed Galal Abdalla Ali

Supervised By:

Dr. Sami Shareef

ABSTRACT

In the last few years, LANs (Local Area Networks) became an essential need for most kind of businesses; therefore the need of having reliable and efficient designs for LANs is an important issue.

This research focuses on creating a new approach for LAN designs aided by Neural Networks. In order to have reliable network design, data (design parameters) have been gathered from working LANs that are considered efficient; this data is then fed to the Neuro-Shell2 software for the purpose of training this software to perform the most efficient design (cabling and network topology) when entering the network requirements to it.

A case study is included in this research to demonstrate the use of Neural Networks to provide the optimum LAN design. This method of design reduces the need of LAN professional designers, but still the need of "fine tuning" is a requirement.

المستخلص

في السنوات القليلة الماضية اصبحت شبكات المناطق المحلية ضرورية لأغلب انواع الأعمال التجارية ، و بالتالي كانت الحاجة الماسّة الى تصميمات فعاله و يمكن الاعتماد عليها.

هذا البحث يركز على ايجاد طريقة جديدة لتصميم شبكاتت المناطق المحلية باستخدام الشبكات العصوبية. للحصول على تصميم ذو اعتمادية عاليه، جمعت البيانات المستخدمة فى هذا البحث من شبكات عامله حاليا و تعتبر ذات كفاءه عالية. هذه البيانات تم تغذية برنامج الشبكات العصوبية بها لانتاج التصميم ذى الكفاءة الأعلى (من ناحية التوصيلات و الطبوغرافيا) عند ادخال المتطلبات من قبل المستخدم.

هنالك دراسة حاله فى هذا البحث لتوضيح استخدام الشبكات العصوبية لانتاج التصميم الأمثل لشبكة المناطق المحلية. أوضحت الدراسات أن هذه الطريقة للتصميم تقلل الحاجة لمصممى الشبكات المحلية المتخصصين، ولكن تبقى اللمسة الأخيرة للمستخدم مهمّه.

Contents

Chapter one: Introduction

1.1. The goal of the project

This project will examine a new methodology for LAN design, in particular backbone cabling and topology. The new methodology uses neural networks and they offer a way of designing a LAN with high precision and significantly lower computational cost than current methods.
Networks can consist of switches, communication devices, etc representing nodes. The neural network can determine the backbone cabling and topology of a LAN with inputs of only the number of nodes, spacing between nodes, and the traffic required between nodes. Artificial neural networks are useful in network design optimization.

1.2. LAN Topologies

Standard models for topologies are the ring, star and bus.

A physical star topology is one in which all branches of the network are connected through a switch. A logical star topology is one in which the switch contains all of the intelligence of the network and directs all network transmissions.

A physical ring topology is one in which all branches of the network are connected to a loop. In a logical ring, data flow from one node to the next in an ordered sequence. When the data reach the last node, they are returned to the originating node.

A physical bus topology connects all networked devices to a single continuous cable. Data may pass directly from one segment to another without first going through a switch or around the ring. In a logical bus topology all network communications are broadcast to the entire network.

A star network is the most reliable, since the remaining segments of the star can still function if one segment goes bad. In a bus or ring topology, a bad segment can bring down the entire network.

1.3. Relationship between cable and topology

The choice of cable and topology are not independent. Table 1.1 shows the preferred combinations. The ring topology requires point-to-point links between repeaters. Twisted-pair wire, baseband coaxial cable, and optical fiber can all be used to provide the links. However, broadband coaxial cable would not work well in this topology. Each repeater would have to be capable of receiving and transmitting data simultaneously on multiple channels.

For the bus topology, twisted pair and both baseband and broadband coaxial cable are appropriate. Until recently, optical fiber cable has not been considered feasible, the multipoint configuration was considered not cost-effective, due to the difficulty in constructing low-loss optical taps. However, recent advances have made the optical fiber bus practical, even at quite high data rates.

The star topology requires a point-to-point link between each device and the central node, most recent activity for this topology has focused on the use of unshielded twisted pair (UTP) over short distances; optical fiber can also be used.

Table 1.1 Cable versus Topology for LANs

Cable	Topology		
	Ring	Bus	Star
Twisted pair	*	*	*
Broadband coaxial cable		*	
Baseband	*	*	

coaxial cable			
Optical fiber	*	*	*
Wireless	*	*	*

1.4. Introduction to Neural Networks

In this section, a brief overview of neural networks will be given.

Neural networks are computational (mathematical) analogs of the basic biological components of a brain; this means NN are just software programs that are designed to operate like the human brain. The software model is based on the connectionist, from the field of Psychology, theorize the brain works. Neural networks are created using software packages such as NeuroShell2. The software creates a topology of nodes (neurons) that are linked together by weighted connections; Figure 1.1 is an example of the topology of a general NN.

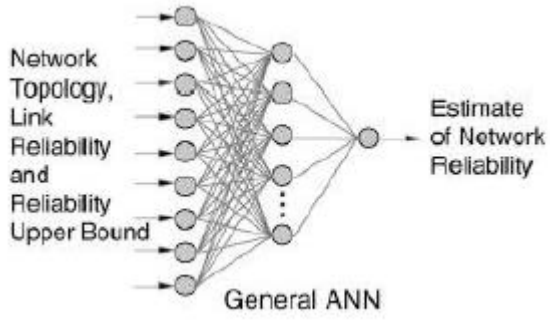


Figure1.1. A graphical representation of a general artificial neural network.

In Fig.1.1 the left column of nodes are the Input nodes. The input nodes receive the input and send signals across weighted links to the middle column of nodes (hidden nodes). The middle column of nodes is called hidden nodes because they receive their input from

other nodes and send their output to other nodes. The last node on the right is the output node, which returns the solutions for the problem.

1.5. General Steps for the Use of Neural Networks in an Application

1. Choose an appropriate neural network model for solving the problem.

2. Prepare data for training the network. This process may include statistical analysis, discretisation and normalization

3. Training the network

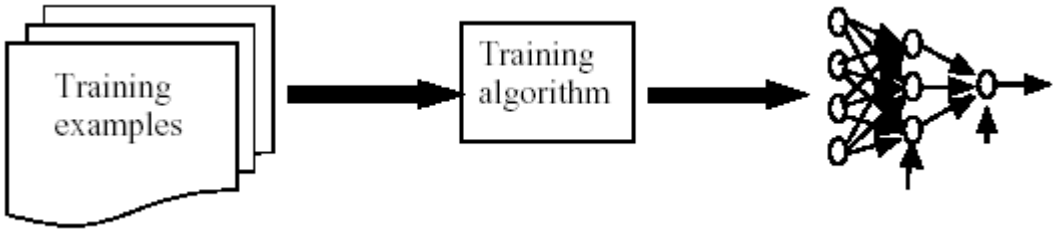


Figure 1.2. Training model

4. Test the network for generalization capability

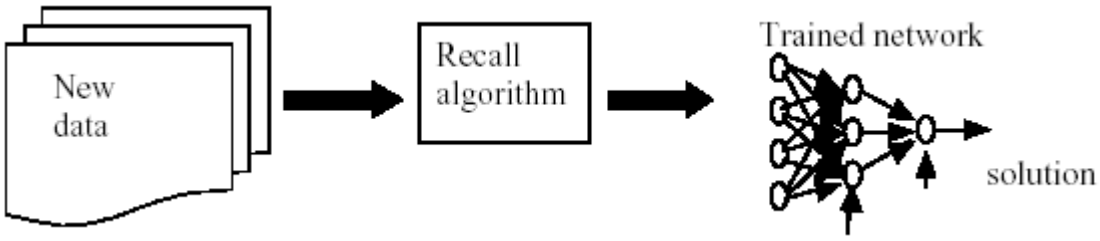


Figure 1.3. Test model

5. Optimize the architecture, if necessary, which may require a repetition of some of the above steps until satisfactory validation results are obtained.

- **Solving problems with neural networks as mapping**

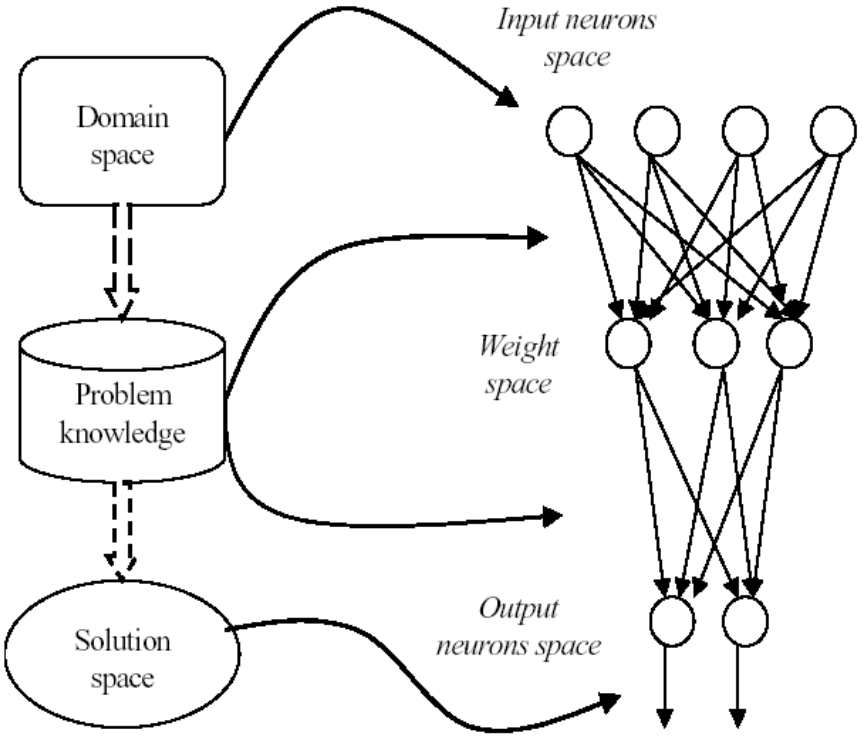


Figure 1.4. Solving problems with neural networks

1.6. Project Outline

First, the problem of optimal LAN design will be discussed in chapter two. In chapter three a more detailed definition and explanation of neural networks, models of a neuron, network architectures, artificial intelligence and neural networks, neural processing, and learning and adaptation, are presented.

In chapter four we tried to present a neural networks application on LAN design using neural networks software package (NeuroShell2), which solve the problem of LAN backbone cabling and topology, this chapter is followed by another chapter containing the results of chapter four in different stages (see section 1.5).

The last chapter (chapter six) is the conclusion of our project.

Chapter 2: LAN Design

2.1. LAN Design Methodology

2.1.1. LAN Design Criteria

Once the questions of why and what LAN? have been answered, the LAN topology and number of segments must be determined. A partial list of factors to consider includes:

- The number of stations.
- The connectivity requirements (hosts, departmental servers).
- The role and location of servers.
- The physical layout of the establishment.
- The existence of affinity groups or any other issue to group stations such as geographical location.
- Performance requirements.
- Reliability and availability (alternate paths, backup gateways).
- Cost per station.
- Systems Management requirements, including server and user software installation, maintenance and distribution.
- Organizational requirements.
- Expected network growth, in the short and long term.
- Consistency and/or intermixing of LAN technology (for example, token-ring and CSMA/CD or FDDI or ATM).

All these factors influence the decision of what topology to select for a particular installation.

Considering the number of factors, it is obvious there is no “best solution” for every network. However, general guidelines can be given on the design methodology and on the location of key functions such as servers.

2.1.2.Key LAN Functions : Servers

During the LAN design process, a planner should decide what type of servers, will be needed and how the users will access them.

There are two categories of servers:

- Central servers.

These servers are usually hosts and might be accessed by all LAN users.
They are usually connected via a backbone and completely managed by the data processing department.

- Local servers.

These servers are primarily connected to a single segment and accessed by a smaller group of related users called an affinity group. Examples of such servers are departmental servers which provide disk sharing or printing facilities.

It is important to decide on the location of the servers, who will maintain them and how they should be managed. These decisions must meet the requirements of both organization and application.

For example, print servers are usually distributed in areas close to their users to provide cost-effective shared printing without time delays or costly print delivery from a central site.
Disk and application servers are often placed in a secure area to avoid accidental damage or intentional misuse. In some environments it may be appropriate to physically group disk servers together, allowing for central management, such as backup, while maintaining a logical relationship to affinity groups.

From a performance standpoint, the load and number of servers should be evaluated to avoid bottlenecks and provide good response time to the LAN users. The probable bottleneck for file servers in a LAN is not the LAN bandwidth but the speed of the DASD and file access methods in the servers.

Finally, a strategy for maintaining software for servers and users must be determined. In particular, if there are many stations, disk or print servers in the installation, automated software maintenance procedures should be considered to alleviate the LAN personnel (or end user) maintenance workload.

2.1.3. Design Methodology

The LAN designer should follow these steps as an iterative methodology to select the "right" LAN topology for a particular installation.

- Collecting the required information such as the type of applications and the number of users of each application, a detailed physical layout "blueprint" and the cabling related information, the number and type of the hosts and servers, performance objectives and traffic statistics, and availability and security requirements.
- User and backbone segment design.
- Backbone scenarios.
- Traffic control between segments.
- Gateway selection.
- Naming convention.
- Network management.
- Migration plan and future growth.

2.2. Logical Design Consideration

Logical design is looking at the relationship between networking components, having little regard for the physical location of these devices.

2.2.1. Logical Design Consideration Using Bridges

Many factors should be considered when designing a LAN consisting of several interconnected LAN segments.

Bridged LAN topology can be influenced by any of the following:

- The number of workstations physically supported by a particular type of segment or cabling.
- Requirements to balance or alleviate heavy traffic associated with particular applications over one or more interconnected LAN segments.
- The MAC protocol used on each segment.
- Requirements for a geographical approach to interconnecting LAN segments by associating segments with specific areas within a building or campus layout or over a telecommunications link through a wide area network.
- Desire to concentrate communications by connecting users with related information needs within LAN segments, known as affinity groups.
- Requirements for performance, reliability and/or availability that may be addressed through use of parallel bridges or parallel routes, providing both increased capacity and backup paths.
- Requirements to separate some LAN stations from others for security purposes by using bridges to provide controllable connectivity paths between secure segments and other stations.
- Requirements to access special function devices such as host gateways or LAN servers, which may best be satisfied through use of a backbone topology.

2.2.2. Multisegment LAN Topologies

This section introduces some basic topologies for interconnected LAN segments including serial, loop, parallel and backbone topologies.

- Serial topology:

This topology may be used in a smaller multisegment network. It is simple, but is usually limited to three segments because a shortcoming of this configuration is that a bridge failure or segment failure will affect overall LAN connectivity. An example of such a topology is shown in Fig.2.1.

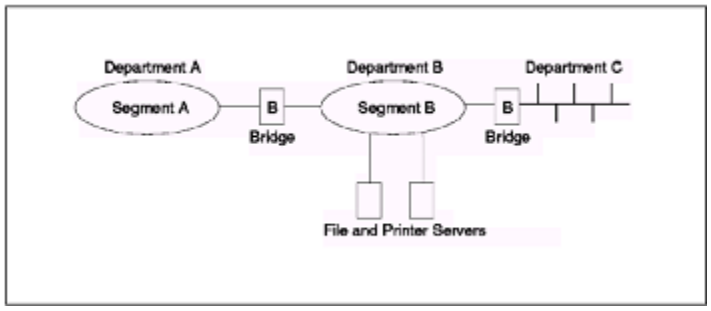


Figure 2.1. Serial LAN Configuration. Segment A and segment C are two separate departments with little inter-communication. They share servers attached to segment B.

- Fully-interconnected and loop configuration

A fully interconnected configuration (mesh) provides alternate paths from each LAN segment to another. Should a bridge or path fail, traffic can be routed through an alternate path, thereby increasing availability of the server segment.

This solution tends to become impractical as the number of LAN segments grows. For n LAN segments, one would require n(n-1)/2 bridges.

Therefore, a loop configuration as shown in Figure 2.2 can be an acceptable compromise between availability versus cost and complexity. As the number of segments increases, however, loop configurations will also develop limitations.

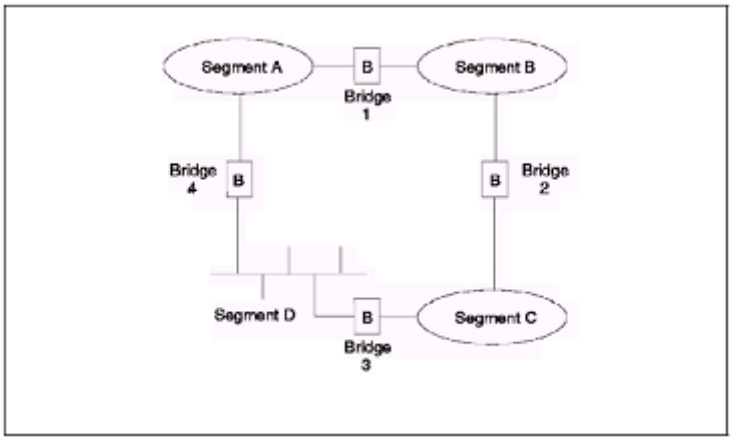


Figure 2.2. Loop Configuration with Four LAN Segments

- Parallel bridge configuration:

Parallel bridges can address the problems of heavy traffic flows through particular bridges and high-availability requirements. While the failure of one bridge would affect connectivity between LAN segments over that bridge, sessions could be recovered via the parallel bridge.

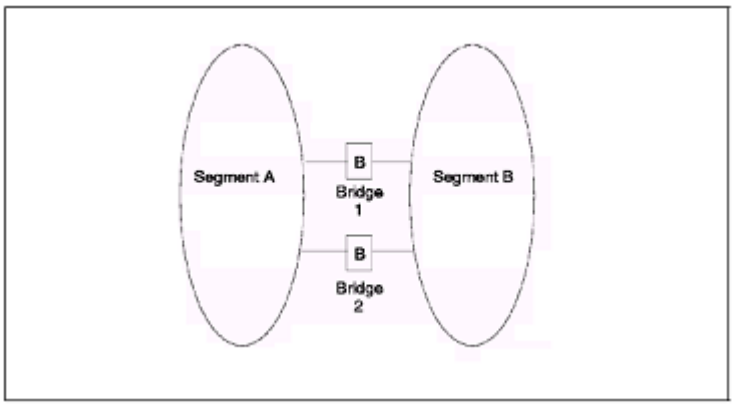


Figure 2.3. Parallel Configuration

The use of active parallel bridges is only possible in a source-routing environment. In a transparent bridging environment, only standby parallel bridges are possible.

- Backbone LAN configurations:

Backbone configurations are usually recommended for large LANs, because they can reduce the number of hops to access common servers and/or gateways from very large numbers of LAN stations.

If growth is an important factor, a backbone LAN configuration will provide the necessary flexibility.

In a backbone configuration, a number of LAN user segments, sometimes known as departmental LANs (or user rings in the case of token-ring segments) are all connected to the same backbone LAN segment as shown in Figure 2.4. This implies that between any two LAN stations attached to departmental LANs, there is always a communications path that includes relatively few bridges, whatever the size of the multisegment LAN.

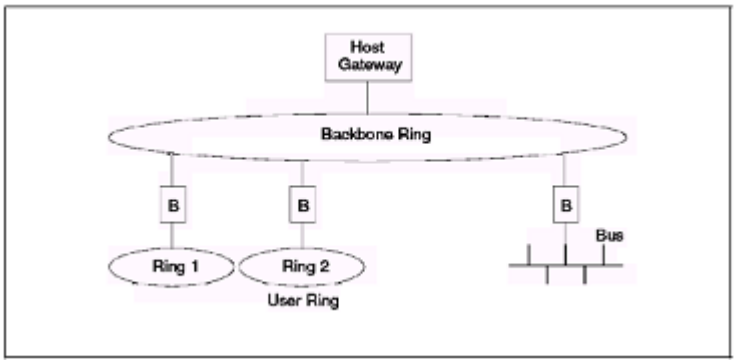


Figure 2.4. Backbone LAN Configuration

2.2.3. User Segments

The designer of user segments (departmental LAN segments) should consider the following factors:

• Physical topology constraints:

The design will usually be influenced by the physical topology of the building. However, distances between the wiring closets, the number of access units, speed and the maximum lobe lengths might sometimes dictate the use of multiple segments per floor, or repeaters, to accommodate the LAN technology physical design rules.

LAN segments are usually implemented by a star-wired ring (IBM Token-Ring) or a star-wired bus (CSMA/CD 10BASE-T). This approach has several advantages over single rings or buses.

-Cabling is easily modified, less expensive cabling such as UTP can be used.

-Defective components (either cabling or stations) are easily detected and isolated.

-The concentrator may have built in repeaters.

-Management functions may be either implemented for standards that do not provide these functions (IEEE 802.3 CSMA/CD is an example), or improved in the case where some management functionality is already provided by the MAC protocol (this is true of the token-ring MAC protocol).

CSMA/CD LANs have physical topology constraints. For example:

The maximum segment lengths for each medium type are as follows:

- 10BASE-5: 500 meters.
- 10BASE-2: 185 meters.
- 10BASE-T: 100 meters.
- 10BASE-F: 2000 meters.

• Number of stations:

The maximum number of stations allowed on a segment varies according to the LAN technology.

A token-ring LAN segment allows up to 260 stations attached on a single ring on STP.

For CSMA/CD LANs, the maximum number of stations on a segment is different for each medium type used:

- 10BASE-5: 100 stations.
-10BASE-2: 30 stations.
-10BASE-T: 2 stations.
-10BASE-F: 2 stations.

• Affinity groups:

An affinity group is a group of users who perform related tasks on the network and have little information interchange with other end users. Each affinity group may be attached to a different segment within a network. This could simplify the design and maintenance of applications running in the servers of the respective affinity groups.

• Organization factors:

Sometimes departments want “their own segment” for management, control or security reasons. Those departments could be completely responsible for selecting and buying the equipment (workstations and servers) and installing and maintaining their local applications.

• Moving (relocation) factor:

If technologies such as wireless LANs are not being used and user groups are often subject to relocation inside a building, use of affinity groups as a basis for LAN structure should be avoided, because the topology may need frequent modification. So if the relocation ratio is high, the design of the user segments should be based essentially on geographical considerations.

• Performance and segment speed:

If client/server or multimedia image-oriented applications are used, very fast high-volume data transfer will be needed to guarantee the service level.

• Management and software maintenance.

2.2.4 High-Availability Design Considerations:

Availability is not the same as reliability. A system component could have a failure rate of once in a hundred years, but that one failure could be tomorrow.

A LAN designer should ask the cost (to the business) of a single LAN failure compared with the cost of the additional components required to improve that availability.

2.2.4.1 Dual Backbone Approach:

Dual backbones, sometimes described as a duplex backbones, address many high-availability requirements and are therefore often recommended for large LANs.

A typical dual backbone configuration is shown in Figure 2.5.

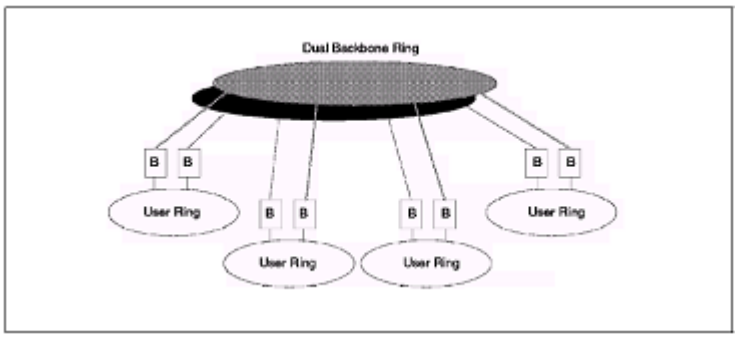


Figure 2.5. Availability Using a Dual Backbone Configuration

Each user segment is connected via two bridges, one to each backbone, although this topology involves more bridges and an additional backbone, there is no single point of failure in the configuration. The major benefits of this topology are:

- If an individual bridge fails, the user will be able to re-establish a session with the partner application via an alternate route.
- Similarly, a backbone failure will not prevent a user from using an alternate route (the other backbone) to communicate with other stations.
- Finally, from a performance standpoint, if connecting token-ring segments with source-routing bridges, the load of the backbones will to some extent be statistically balanced during session establishment because of the source-routing algorithm.

2.3.physical Design Consideration

At this point of the network design process, a number of possible logical topologies will have been created, but the functional components they contain will need to be mapped onto real devices, into the real physical infrastructure within a building or campus. The whole process will be iterative, with no single correct answer. Eventually a complete network system design will be created, with appropriate detailed documentation and plans, which then can be implemented.

2.3.1 Recommendations for Horizontal Cabling

Horizontal cabling is the cable that runs from telecommunications closets to offices or work areas.The cabling choices provided by the standards include:

150W shielded twisted-pair (STP) cable (two-pair), 22 gauge, types 1, 1A, 2, and 2A.
100W unshielded twisted-pair (UTP) cable, Categories 3, 4, and 5 (two and four pair).
100W twisted-pair cable with an overall shield, Categories 3, 4, and 5 (two and four pair).

120W twisted-pair cable with an overall shield - primarily Category 5 (two and four pair).
62.5/125-micron optical fiber (50/125-micron cable is an allowed option).

2.3.2. Recommendations for Building and Campus Backbone Cabling

IBM supports the use of both multi-mode and single-mode optical fiber in building and campus backbone applications. Not all applications are supported on single-mode and multi-mode optical fiber. Other cables and connectors are also supported, but support may be at reduced distances. Copper cabling, both 150W STP or STP-A, and Category 5 UTP may be appropriate, generally for inter-telecommunications closet distances not exceeding 90 meters, and within a single building. However, this copper cabling should be a supplement to, and not a substitute for, the recommended optical fiber cable interconnecting the telecommunications closets in a campus network.

2.3.3. Standard, (TIA/EIA-568-A)

The purpose of the standard is to enable planning and installation of a structured cabling system for commercial buildings for generic purposes. Especially for horizontal cabling, the cable between a wiring closet and a work area outlet, installation of a cabling system during building construction or renovation is significantly less expensive and less disruptive than after a building is occupied.

The document covers cabling plans to the order of 3,000m in extent, with populations to the order of 50,000 users.

The cabling model is shown in Figure 2.6 on page 17. It is not an all-inclusive representation, but is meant as a typical example. For example, in a real implementation it is probably better to have two wiring closets (TCs) supporting a given floor area, some users using one, others the other.

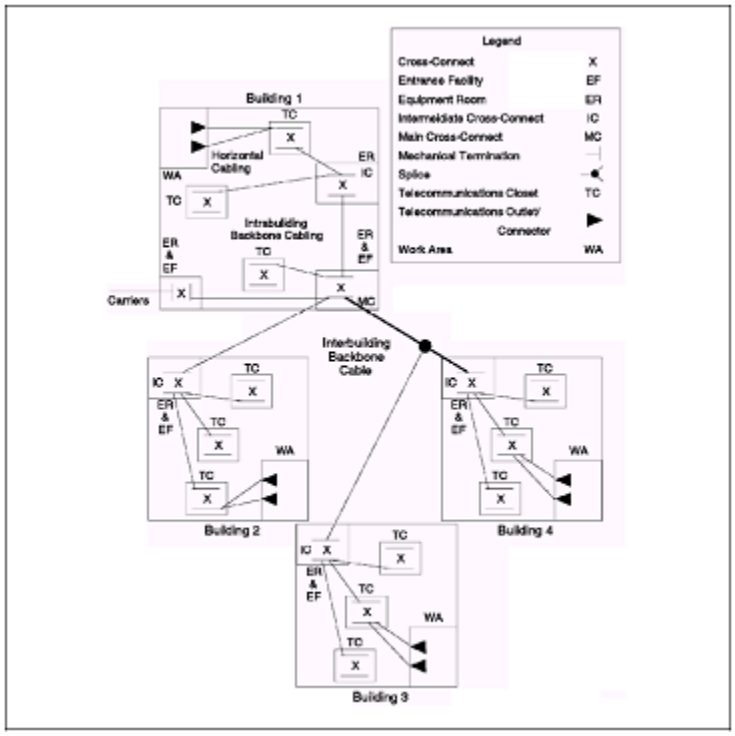


Figure 2.6. The EIA/TIA Structured Cabling Model

The horizontal cabling is the portion of the cabling system that extends from the work area outlet to the horizontal cross connects in the wiring closet. The horizontal cabling contains the greatest quantity of individual cables within a building, and after construction, is least accessible for change and most disruptive to users during change. Of all the components in a cabling system, the horizontal cabling is the component that should be “done right” the first time.

The horizontal cable is:

- Continuous, that is, there are no splices or bride taps (used for telephone extensions); the only join allowed is between an under-carpet (flat) cable and a round horizontal cable.
- A maximum of 90 m in length. The standard then allows for 10 m of other interconnect cable, patch leads, cross connects and so forth (3 m device to outlet is assumed).

The allowed cable types are:

- Four-pair 100□UTP (category 3-5).
- Two-pair STP-A
- Two-core 62.5/125 □m fiber.

The backbone cabling provides interconnection between the telecommunication closets and equipment rooms in a star-wired configuration. The standard allows telecommunications closets to be wired together for bus and ring topologies.

From a data communications perspective, this is almost prerequisite for any sensible use of an installed cabling system; such interconnects are pointless for telephones. This is shown schematically in Figure 2.7.

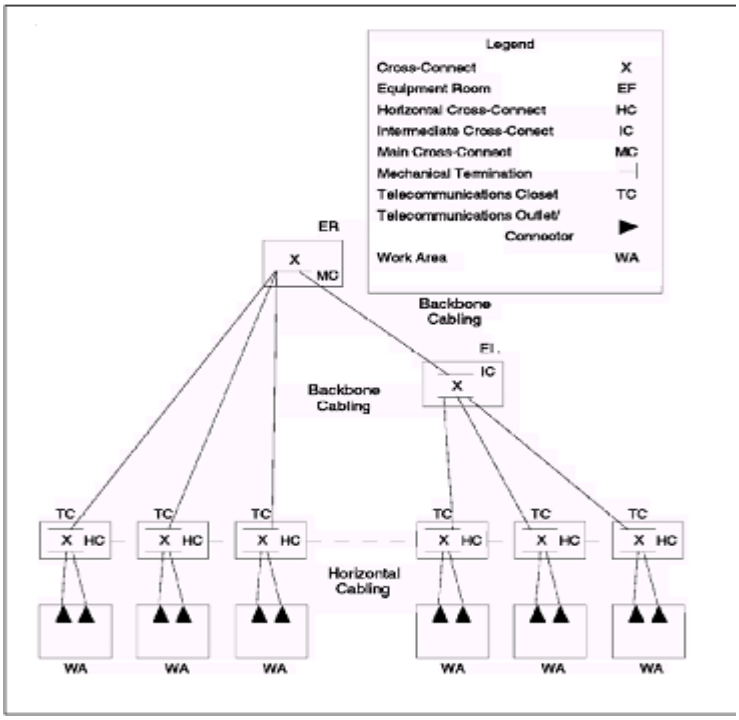


Figure 2.7. Backbone Cabling Hierarchical Star Topology

The recognized media for the backbone cables are:

- 100ΩUTP category 3-5.
- 150ΩSTP-A.
- 62.5/125 μmfiber cable.

- Single-mode fiber cable.

The advised cable lengths are:

- 90 m for copper.
- 2000 m total fixed cable for multi-mode fiber (horizontal-main cross or horizontal-intermediate-main cross-connect).
- 3000 m total fixed cable for single mode fiber.

2.3.4. The SC Connector

Just as you thought the world was standardizing on the ST fiber connector, the standards organizations have moved on.

The draft amendment to the EIA/TIA 568 Commercial Building Telecommunications Cabling Standard, proposal Number 2840-A, dated 4/29/94 is recommending the SC connector for all future building fiber cabling. When approved, this standard will be published as EIA/TIA 568-A.

The EIA connector is for 62.5/125 mm and single-mode optical fiber cabling. The specification is for a keyed duplex connector for use within the structured cabling system (backbone, horizontal cabling, main cross connect, intermediate crass connects, horizontal cross connects and the telecommunications outlet and connector).

The standard does allow for users of ST connectors to continue, but any new installations should use SC connectors.

The 568SC connector is a pair of small keyed square connectors mounted adjacent to each other, marked “A” and “B.” Single-mode components are blue and the multi-mode are beige.

2.3.5. Ethernet Single Collision Domain Physical Design Rules

Ethernet (IEEE 802.3) used the CSMA/CD access method. The collision detect function (CD) requires the physical span of the collision domain to be contained. If two devices at either end of a collision domain start transmitting at the same time, either device needs to be aware a collision has occurred before they finish transmitting. The dynamics of the collision handling mechanism are largely determined by a single defined parameter, the slot time. For Ethernet, the slot time is 512 bit-times (64 bytes). The total round-trip delay in the network must be less than the slot time.

The total round-trip delay in a repeated Ethernet network is the sum of parameters such as:

- The propagation delay in the Ethernet cable.
- Delays incurred in the repeaters.
- DTE delays.
- Transceiver delays.

The IEEE also prescribes a maximum length for each media type:

- 10BASE-5: 500 m (maximum).
- 10BASE-2: 185 m (maximum).
- 10BASE-T: up to at least 100 m.
- FOIRL: 1000 m (maximum).
- 10BASE-FL: 2000 m.
- 10BASE-FB: 2000 m.

The 10BASE-T length is a minimum, but it is really based on the signal loss in dBs, (11.5 dB maximum loss source to destination).
The fiber lengths depend on the signaling technology, fiber budget and medium used.

There are a number of other considerations, including:
• The maximum number of stations allowed on a segment varies according to the type of medium used:
– 10BASE-5: 100 stations.

– 10BASE-2: 30 stations.

– FOIRL: 2 stations.

– 10BASE-T: 2 stations.

– 10BASE-FL: 2 stations.

– 10BASE-FB: 2 stations.

• The maximum number of stations in a collision domain is 1024.

• Repeaters can be attached at any position on a coax segment but should be on the end of link segments.

• 10BASE-2 segments should not be used to interconnect 10BASE-5 segments.

• Each repeater takes one attachment position on the segment and should be counted towards the maximum number of stations allowed on that medium.

2.3.6. Physical Design Summary

A finalized network design can be created after a few iterations of:

- Logical design.
- Physical design.
- Design reviews and simulations.

This can be as detailed as an organization requires it to be, from simple sketches of device locations to detailed installation plans defining an Installation down to the last nut and bolt.

CHAPTER THREE: NEURAL NETWORKS

3.1. What Is a Neural Network?

In its most general form, a neural network is a machine that is designed to model the way in which the brain performs a particular task or function of interest; the network is usually implemented by using electronic components or is simulated in software on a digital computer.

To achieve good performance, neural networks employ a massive interconnection of simple computing cells referred to as "neurons" or "processing units." We may thus offer the following definition of a neural network viewed as an adaptive machine:

A neural network is a massively parallel-distributed processor made up of simple processing units, which has a natural propensity for storing experiential knowledge and making it available for use. It resembles the brain in two respects:

1. Knowledge is acquired by the network from its environment through a learning process.
2. Interneuron connection strengths, known as synaptic weights, are used to store the acquired knowledge.

The procedure used to perform the learning process is called a learning algorithm, the function of which is to modify the synaptic weights of the network in an orderly fashion to attain a desired design objective.

3.1.1. Benefits of Neural Networks

It is apparent that a neural network derives its computing power through, first, its massively parallel distributed structure and, second, its ability to learn and therefore generalize. Generalization refers to the neural network producing reasonable outputs for inputs not encountered during training (learning).

In practice, however, neural networks cannot provide the solution by working individually. Rather, they need to be integrated into a consistent system engineering approach. Specifically, a complex problem of interest is decomposed into a number of relatively simple tasks, and neural networks are assigned a subset of the tasks that match their inherent capabilities.

The use of neural networks offers the following useful properties and capabilities:

1. Nonlinearity.
2. Input-Output Mapping.
3. Adaptivity.
4. Evidential Response.
5. Contextual Information.
6. Fault Tolerance.
7. VLSI Implementation.
8. Uniformity of Analysis and Design.
9. Neurobiological Analogy.

3.2. MODELS OF A NEURON

A neuron is an information-processing unit that is fundamental to the operation of a neural network. The block diagram of Fig. 3.1 shows the model of a neuron, which forms the basis for designing (artificial) neural networks.

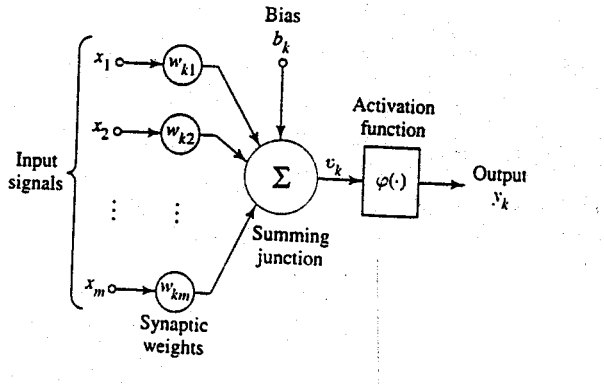


FIGURE 3.1 Nonlinear model of a neuron

Here we identify three basic elements of the neuronal model:

1. A set of synapses or connecting links, each of which is characterized by a weight or strength of its own. Specifically, a signal Xj at the input of synapse j connected to neuron k is multiplied by the synaptic weight Wkj.
2. An adder for summing the input signals, weighted by the respective synapses of the neuron; the operations described here constitute a linear combiner:
3. An activation function for limiting the amplitude of the output of a neuron. The activation function is also referred to as a squashing function in that it squashes (limits) the permissible amplitude range of the output signal to some finite value.

The bias b $_k$ has the effect of increasing or lowering the net input of the activation function, depending on whether it is positive or negative, respectively.

In mathematical terms, we may describe a neuron k by writing the following pair of equations:

$$U_k = \sum_{j=1}^{m} W_{kj} - X_j$$

And

$$Y_k = \Phi(U_k + b_k)$$

where X_1, X_2... X_m are the input signals; W_{k1}, W_{k2}... W_{km} are the synaptic weights of neuron k; U_k is the linear combiner output due to the input signals; b_k is the bias; Φ (.) is the activation function; and Y_k is the output signal of the neuron.

3.2.1. Types of Activation Function

The activation function, denoted by $\Phi(v)$, defines the output of a neuron in terms of the induced local field v. Here we identify three basic types of activation functions:

1. Threshold Function. For this type of activation function, described in Fig.3.2a, we have

$$\Phi(v) = \begin{matrix} 1 \text{ if } v \geq 0 \\ 0 \text{ if } v < 0 \end{matrix}$$

In engineering literature, this form of a threshold function is commonly referred to as a Heaviside function. Correspondingly, the output of neuron k employing such a threshold function is expressed as

$$Y_k = \begin{matrix} 1 \text{ if } v_k \geq 0 \\ 0 \text{ if } v_k < 0 \end{matrix}$$

Where V_k is the induced local field of the neuron; that is,

$$V_k = \sum_{j=1}^{m} W_{kj} X_j + b_k$$

2. Piecewise-Linear Function. For the piecewise-linear function described in fig.3.2b we have

$$\Phi(v) = \begin{cases} 1, & v \geq +1/2 \\ v, & +1/2 > v > -1/2 \\ 0, & v \leq -1/2 \end{cases}$$

Where the amplification factor inside the linear region of operation is assumed to be unity. This form of an activation function may be viewed as an approximation to a nonlinear amplifier. The following two situations may be viewed as special forms of the piecewise-linear function:

- A linear combiner arises if the linear region of operation is maintained without running into saturation.
- The piecewise-linear function reduces to a threshold function if the amplification factor of the linear region is made infinitely large.

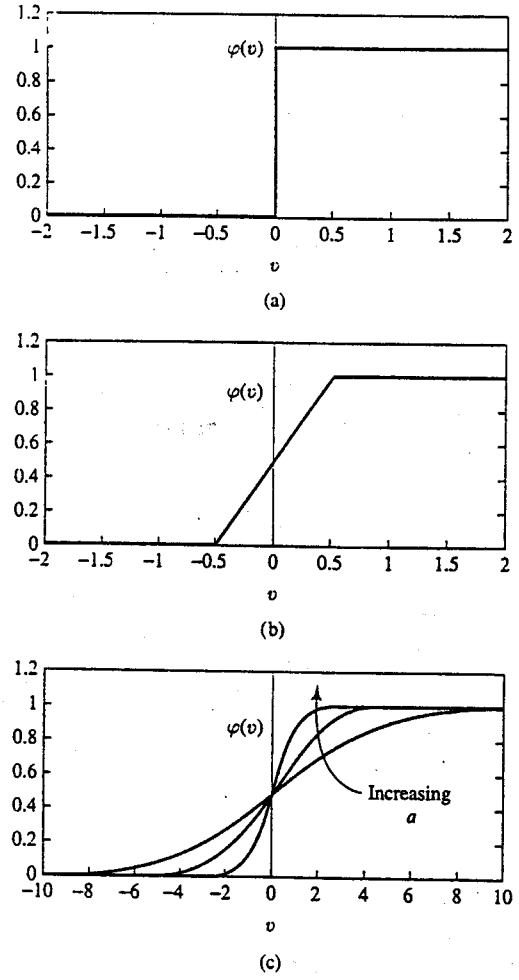


FIGURE 3.2 (a) Threshold function, (b) Piecewise-Linear Function,
(c)Sigmoid Function for varying slope parameter a

3. Sigmoid Function. The sigmoid function, whose graph is s-shaped, is by far the most common form of activation function used in the construction of artificial neural networks. It is defined as a strictly increasing function that exhibits a graceful balance between linear and nonlinear behavior. An example of the sigmoid function is the logistic function, defined by

$$\Phi(v) = (1 + \exp(-av/t))^{-1}$$

Where **a** is the slope parameter of the sigmoid function. By varying the parameter a, we obtain sigmoid functions of different slopes, as illustrated in Fig.3.2c.

3.3. Network Architectures

In general, we may identify three fundamentally different classes of network architectures:

1. Single-Layer Feedforward Networks

In a layered neural network the neurons are organized in the form of layers. In the simplest form of a layered network, we have an input layer of source nodes that projects onto an output layer of neurons (computation nodes), but not vice versa. In other words, this network is strictly a feedforward or acyclic type. It is illustrated in Fig. 3.3 for the case of four nodes in both the input and output layers. Such a network is called a single-layer network, with the designation "single-layer" referring to the output layer of computation nodes (neurons).

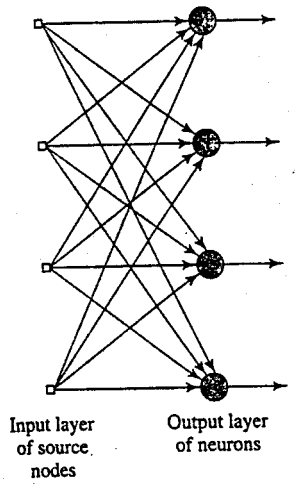


FIGURE 3.3 Feedforward or acyclic network with a single layer of neurons

2. Multilayer Feedforward Networks

The second class of a feedforward neural network distinguishes itself by the presence of one or more hidden layers, whose computation nodes are correspondingly called hidden neurons or hidden units. The function of hidden neurons is to intervene between the external input and the network output in some useful manner. By adding one or more

hidden layers, the network is enabled to extract higher-order statistics. In a rather loose sense the network acquires a global perspective despite its local connectivity due to the extra set of synaptic connections and the extra dimension of neural interactions. The ability of hidden neurons to extract higher-order statistics is particularly valuable when the size of the input layer is large.

The source nodes in the input layer of the network supply respective elements of the activation pattern (input vector), which constitute the input signals applied to the neurons (computation nodes) in the second layer (i.e., the first hidden layer). The output signals of the second layer are used as inputs to the third layer, and so on for the rest of the network. Typically the neurons in each layer of the network have as their inputs the output signals of the preceding layer only. The set of output signals of the neurons in the output (final) layer of the network constitutes the overall response of the network to the activation pattern supplied by the source nodes in the input (first) layer.

The neural network in Fig. 3.4 is said to be fully connected in the sense that every node in each layer of the network is connected to every other node in the adjacent forward layer. If, however, some of the communication links (synaptic connections) are missing from the network, we say that the network is partially connected.

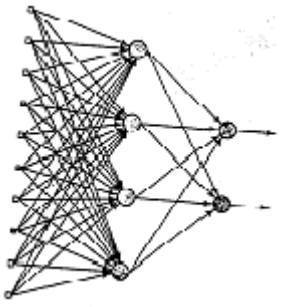


FIGURE 3.4 Feedforward or acyclic network with one hidden layer and one output layer

3. Recurrent Networks

A recurrent neural network distinguishes itself from a feedforward neural network in that it has at least one feedback loop. For example, a recurrent network may consist of a single layer of neurons with each neuron feeding its output signal back to the inputs of all the other neurons, as illustrated in the architectural graph in Fig. 3.5. In the structure depicted in this figure there are no self-feedback loops in the network; self- feedback refers to a situation where the output of a neuron is fed back into its own input. The recurrent network illustrated in Fig. 3.5 also has no hidden neurons.

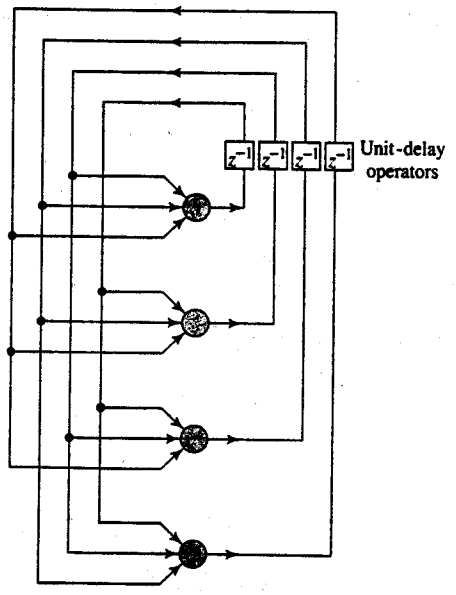


FIGURE 3.5 Recurrent Networks with no self-feedback loops and no hidden neurons

3.4. Artificial Intelligence and Neural Networks

An AI system must be capable of doing three things: (1) store knowledge, (2) apply the knowledge stored to solve problems, and (3) acquire new knowledge through experience. An AI system has three key components: representation, reasoning, and learning, as depicted in Fig. 3.6.

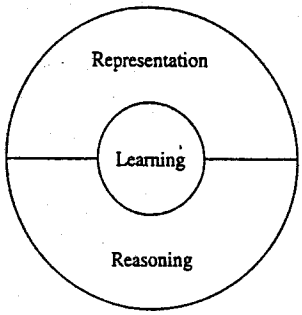


FIGURE 3.6 illustrating the three key components of an AI system

1. Representation: The most distinctive feature of AI is probably the pervasive use of a language of symbol structures to represent both general knowledge about a problem domain of interest and specific knowledge about the solution to the problem.

2. Reasoning: In its most basic form, reasoning is the ability to solve problems. For a system to qualify as a reasoning system it must satisfy certain conditions:

- The system must be able to express and solve a broad range of problems and problem types.
- The system must be able to make explicit and implicit information known to it.
- The system must have a control mechanism that determines which operations to apply to a particular problem, when a solution to the problem has been obtained, or when further work on the problem should be terminated.

3. Learning: In the simple model of machine learning depicted in Fig. 3.7, the environment supplies some information to a learning element. The learning element then uses this information to make improvements in a knowledge base, and finally the performance element uses the knowledge base to perform its task. The kind of information supplied to the machine by the environment is usually imperfect, with the result that the learning element does not know in advance how to fill in missing details or to ignore details that are unimportant. The machine therefore operates by guessing,

and then receiving feedback from the performance element. The feedback mechanism enables the machine to evaluate its hypotheses and revise them if necessary.

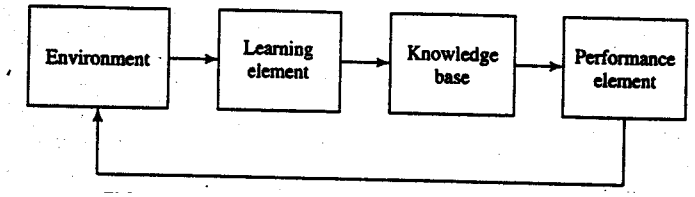


Figure 3.7 Simple model of machine learning

3.5. Neural Processing

The process of computation of **0** for a given **x** performed by the network is known as *recall.* Recall is the proper processing phase for a neural network, and its objective is to retrieve the information. Recall corresponds to the decoding of the stored content, which may have been encoded in a network previously.

Assume that a set of patterns can be stored in the network. Later, if the network is presented with a pattern similar to a member of the stored set, it may associate the input with the closest stored pattern. The process is called *autoassociation.* Typically, a degraded input pattern serves as a cue for retrieval of its original form. This is illustrated schematically in Fig 3.8(a).

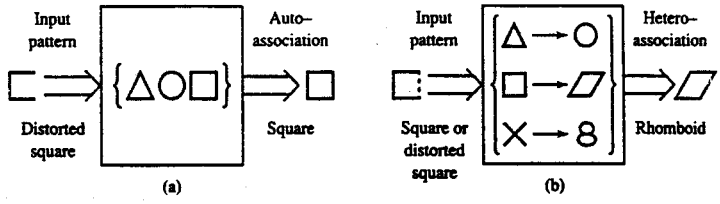


Figure 3.8 Association response (a) autoassociation and (b) hetroassociation

Associations of input patterns can also be stored in a *heteroassociation* variant. In heteroassociative processing, the associations between pairs of patterns are stored. This is schematically shown in Figure 3.8(b). A square input pattern presented at the input results in the rhomboid at the output. It can be inferred that the rhomboid and square constitute one pair of stored patterns. A distorted input pattern may also cause correct heteroassociation at the output as shown with dashed line.

Classification is another form of neural computation. Let us assume that a set of input patterns is divided into a number of classes, or categories. In response to an input pattern from the set, the classifier is supposed to recall the information regarding class membership of the input pattern. Typically, classes are expressed by discrete-valued output vectors, and thus output neurons of classifiers would employ binary activation functions. The schematic diagram illustrating the classification response for patterns belonging to three classes is shown in Figure 3.9(a).

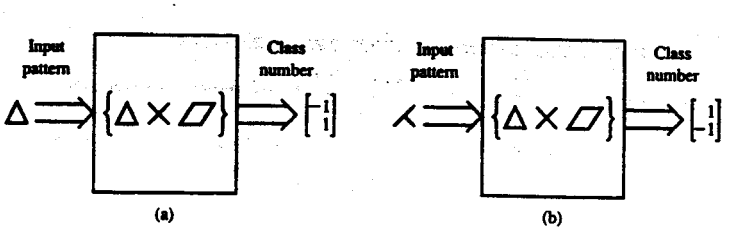


Figure 3.9 Classification responses :(a) Classification and (b) recognition

If the network's desired response is the class number but the input pattern does not exactly correspond to any of the patterns in the set, the processing is called *recognition.* When a class membership for one of the patterns in the set is recalled recognition becomes identical to classification. Recognition within the set of three patterns is schematically shown in Figure 3.9(b). This form of processing is of particular significance when an amount of noise is superimposed on input patterns.

One of the distinct strengths of neural networks is their ability to generalize. The network is said to generalize well when it sensibly interpolates input patterns that are new to the network. Assume that a network has been trained using the data x_1 through x_5 as shown in Figure 3.10. The figure illustrates bad and good generalization examples at

points that are new and are between the training points. Neural networks provide. in many cases, input-output mappings with good generalization capability.

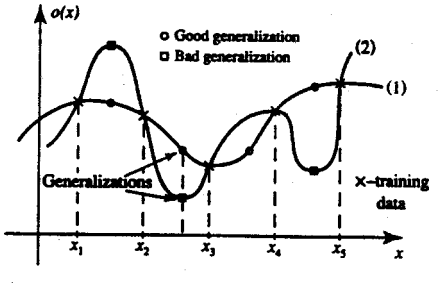


Figure 3.10 Generalization example

3.6. Learning and Adaptation

Learning in neural networks is a more direct process, and we typically can capture each learning step in a distinct cause-effect relationship. To perform any of the processing tasks discussed in the previous section, neural network learning of an input-output mapping from a set of examples is needed. Designing an associator or a classifier can be based on learning a relationship that transforms inputs into outputs given a set of examples of input-output pairs.

3.6.1. Learning as Approximation or Equilibria Encoding

Approximation theory focuses on approximating a continuous, multivariable function h(x) by another function $H(w,$ x), where x = $[x_1 \quad x_2 \ \ldots \ x_n]$ is the input vector and w = $[w_1 \ \ w_2 \ \ldots w_m]$ is a parameter (weight) vector. In the approach below, we will look at a class of neural networks as systems that can learn approximation of relationships. The learning task is to find w that provides the best possible approximation of h(x) based on - the set of training examples {x}. An important choice that needs to be made is which approximation function H (w, X) to use. An ill-chosen, nonsmooth, approximation

function example is shown in Figure 3.10 as curve (2). Even with the best choice of parameters for an ill-chosen function, the approximation is inaccurate between successive data points. The choice of function *H (w, x)* in order to represent h(x) is called a *representation* problem. Once H (w, x) has been chosen, the network learning algorithm is applied for finding optimal parameters w.

3.6.2. Supervised and Unsupervised Learning

In *supervised learning* we assume that at each instant of time when the input is applied, the desired response d of the system is provided by the teacher. This is illustrated in Figure 3.11(a). The distance *p* [*d*, 0] between the actual and the desired response serves as an error measure and is used to correct network parameters externally. Since we assume adjustable weights, the teacher may implement a reward-and-punishment scheme to adapt the network's weight matrix W. For instance, in learning classifications of input patterns or situations with known responses, the error can be used to modify weights so that the error decreases. This mode of learning is very pervasive. Also, it is used in many situations of natur3llearning. A set of input and output patterns called a *training set* is required for this learning mode.

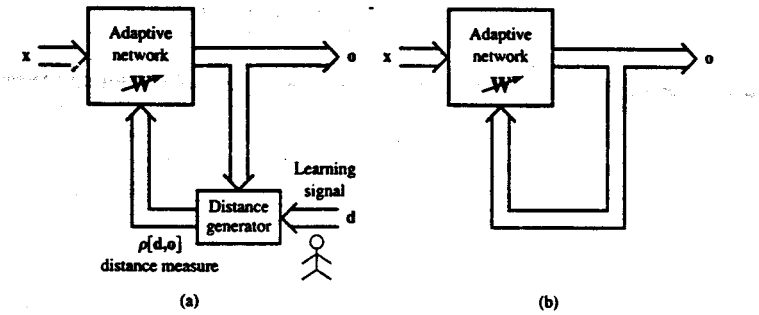


Figure 3.11 Basic learning mode: (a) supervised learning and (b) unsupervised learning

Figure 3.11(b) shows the block diagram of unsupervised learning. In *learning without supervision,* the desired response is not known; thus, explicit error information cannot be used to improve network behavior. Since no information is available as to correctness or

incorrectness of responses, learning must somehow be accomplished based on observations of responses to inputs that we have marginal or no knowledge about.

3.7. Overview of Neural Networks

More than a dozen specific architectures are covered in this text. Let us attempt to group them into classes. There seem to be numerous ways of classifying artificial neural systems for the purposes of study, analysis, understanding, and utilization.

Table 3.1 summarizes the taxonomy of the most important artificial neural system architectures. Thirteen different architectures are listed in the first column of the table. Only basic network configurations covered in this text are included, therefore the table is by no means exhaustive.

TABLE 3.1 Classification of the most important ar1ificial neural networks according to their learning and recall modes.

Network Architecture	Learning mode S,U,R	Recall mode FF,REC	Recall time domain CT, DT
Single-layer Network of Discrete and continuous Perceptions	S	FF	_
Multilayer Network of Discrete and Continuous Pen:eptrons	S	FF	_
Gradient-type Network	R	REC	CT

Linear Associative Memory	R	FF	_
Autoassociative Memory	R	REC	DT or CT
Bidirectional Associative Memory	R	REC	DT or CT
Temporal Associative Memory	R	REC	DT or CT
Hamming Network	R	FF	_
MAXNET	R (fixed)	REC	DT or CT
Clustering Network	U	FF	_
Counterpropagation Network	U+S	FF	_
Neural Array	U	FF	_
Adaptive Resonance Theory I Network	U	REC	DT or CT

Learning Mode
S-Supervised
U-Unsupervised
R-Recording (Batch)

Recall mode
FF-Feedforward
REC-Recurrent

Recall Time domain (only for recurrent networks)
CT -Continuous-time
DT-Discrete-time

The learning and recall modes are the most important characteristics of the networks we study. Supervised / unsupervised / recording (batch) learning modes are specified in the table for each network of interest. Feedforward, or recurrent recall mode, is also highlighted. Note that feedforward networks provide instantaneous recall. Additionally, continuous- and discrete-time networks have been marked within the group of recurrent networks.

Neural networks can also be characterized in terms of their input and output values as discrete (binary) or continuous. Diverse neuron models are used in various networks, sometimes even a combination of them.

Chapter four: Neural Networks Application on LAN Design

4.1. Introduction

In this chapter we shall consider a case study of LAN design using Neural Networks Software Packet, NeuroShell2.

Data is containing LAN designs with different BW requirements, number of nodes, spacing between nodes, cabling and topology. This data is then fed to the 'NeuroShell2' in order to enable it create a neural network that can be used to predict the optimum solution for a LAN topology and cabling.

It is worthwhile noting that although using Neural Networks can suggest to the designer the network topology and cabling that fulfills the LAN requirements, however, matters such as scalability and possibility of network expansion are important factors to be taken into consideration by the designer before applying Neural Networks in the data gathering stage.

4.2. Case Study

The case we are considering to apply NeuroShell2 on is a backbone NW. Assuming that the LAN have been properly segmented into several collision domains (segments) using LAN switches, now comes the task of interconnecting these switches to form a backbone LAN.

There are several valid topologies and cabling types that can be deployed in such a network, these vary according to several factors or parameters (that are discussed in the following section). In our case study we takes only three options of topologies: Bus, Star and Ring, and two options of cabling: UTP cat5 and optical fiber single mode (62.5 μm).

4.2.1. Backbone NW

A backbone network primary task is to switch data as fast as possible between the switches that are participating in that network.

The speed requirement is due to the fact that the backbone links are to carry traffic between different segments of the LAN as it is shown in fig (2.1), (2.2) and (2.3), backbone network is described also in chapter2, section 2.2.2.

4.2.2 Backbone NW Topologies

- **Linear bus backbone**

Using a linear bus configuration for the backbone network requires a separate backbone cable to which backbone access devices are connected. These devices also connect to their respective LANs, as shown.

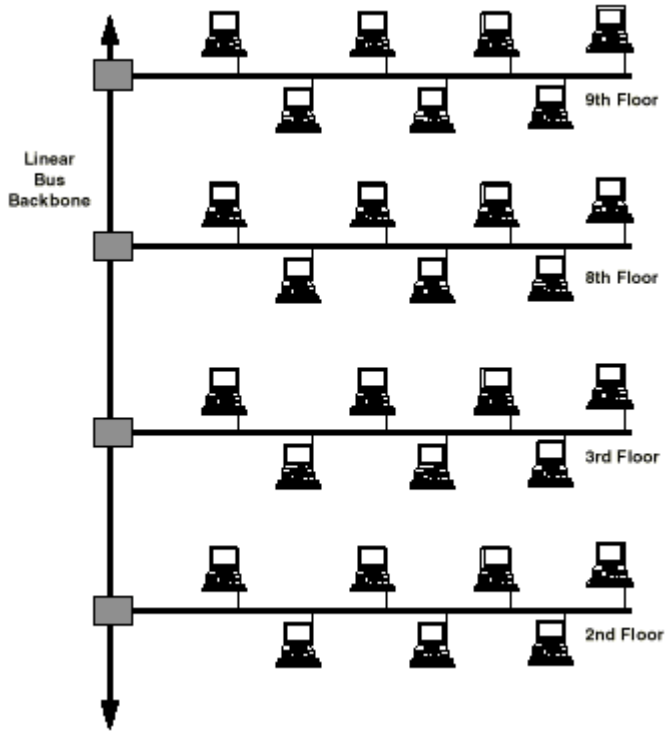


Figure 4.1: Linear bus backbone network

- **Ring backbone**

In a ring topology, the devices used to access the backbone are connected to each other in series.

The last device is then connected to the first, forming the ring.

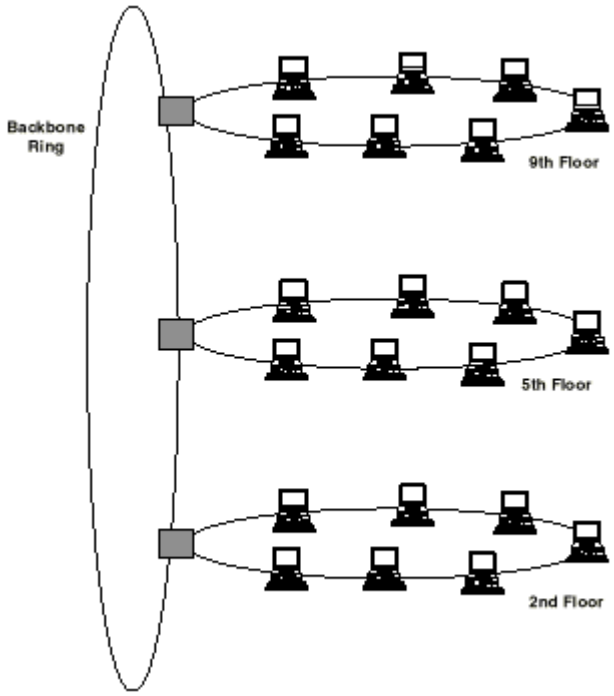


Figure 4.2. Ring backbone network

- **Hierarchical star backbone**

The hierarchical star is the topology recommended by ANSI/TIA/EIA-568-A. In such a configuration, the devices connecting to the LANs are also connected to a central device, forming the star.

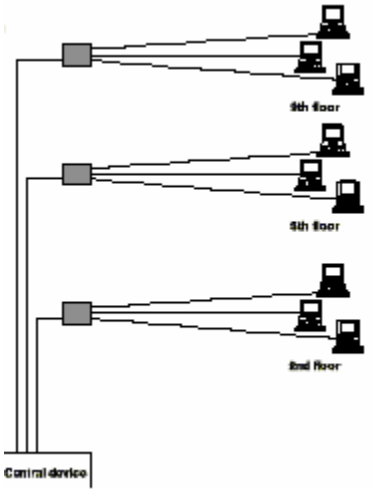


Figure 4.3. Hierarchical star backbone

4.3. Applying NeuroShell 2

4.3.1 Data gathering

The type of cabling used in a backbone network is dependent on BW requirements, geographical location of nodes (spacing), number of taps and of course the network topology.

Tables4.1 and 4.2 provide illustration of the cabling types we are considering in our case.

Table 4.1 using UTP

Topology	BW	Spacing	No of taps
RING	4 Mbps	0.1km (repeater spacing)	70
STAR	10-100 Mbps	0.1km (node to central device)	105
BUS	1 Mbps	<2km	105

Table 4.2 using Optical Fiber

Topology	BW	Spacing	No of taps
RING	100 Mbps	2km (repeater spacing)	250
STAR	100 Mbps	0.1 km	105
		1.8 km	32
BUS	45Mbps	<150 km	500taps

From the two tables above we can design several LANs with different number of nodes, nodes spacing, traffic between nodes, cabling and topologies as shown in table 4.3.

Table 4.3 LAN design examples

Traffic (In Mbps)	Nodes number	Nodes spacing (In Km)	Topology	Cabling
100	2	0.10	Bus	UTP
200	2	0.50	Bus	UTP
400	2	1.00	Bus	Optical Fiber
800	2	2.00	Bus	Optical Fiber
1000	2	5.00	Bus	Optical Fiber
100	5	0.10	Star	UTP
200	5	0.20	Star	UTP
400	5	0.30	Ring	UTP
800	5	0.40	Ring	UTP
1000	5	0.50	Ring	Optical Fiber
100	8	0.10	Star	UTP
200	8	0.45	Star	UTP
400	8	1.20	Star	Optical Fiber
800	8	3.00	Ring	Optical Fiber
1000	8	9.00	Ring	Optical Fiber
100	10	1.00	Star	UTP
200	10	2.00	Star	UTP
400	10	4.00	Ring	UTP
800	10	10.00	Ring	Optical Fiber
1000	10	19.00	Ring	Optical Fiber
100	12	5.00	Star	Optical Fiber
200	12	11.00	Ring	Optical Fiber
400	12	21.00	Ring	Optical Fiber
800	12	48.00	Ring	Optical Fiber
1000	12	77.00	Ring	Optical Fiber

100	15	1.50	Star	UTP
200	15	4.30	Star	UTP
400	15	10.00	Ring	Optical Fiber
800	15	18.00	Ring	Optical Fiber
1000	15	33.00	Ring	Optical Fiber
100	20	0.50	Ring	UTP
200	20	0.90	Ring	UTP
400	20	2.10	Star	Optical Fiber
800	20	8.00	Ring	Optical Fiber
1000	20	24.00	Ring	Optical Fiber

The data in table 4.3 are fed to the software through the datagrid, which is the spreadsheet designed in NeuroShell 2 for use with small files to enter data directly into NeuroShell 2 format. This data are used as training patterns for the neural network.

Neural networks cannot process data that are not numeric. For this reason we can use Symbol Translate module in NeuroShell 2 to translate text into numeric values that the network capable of processing. We translate ring to “1”, star to “2”, bus to “3”, UTP to “10” and optical fiber to “20”.

4.3.2. Build Neural Network

4.3.2.1. Define Inputs and Outputs

First we used the Define Inputs/Outputs module to choose which variables will be used as network inputs and outputs, and to compute the minimum and maximum values for each variable.

4.3.2.2. Test Set Extraction

Then the Test Set Extract module is used to extract a test set of data form the training patterns. The test set used with calibration to prevents overtraining networks so they will generalize well on new data. It is also used to test the network's results with data the network has never seen before.

4.3.2.3. Network Design

General Regression Neural Networks (GRNN) are known for their ability to train quickly on sparse data sets and it is applications are able to produce continuous valued outputs. In our tests we found that GRNN responds much better than other networks to our case study. GRNN is a type of supervised network.

GRNN is a three-layer network where there must be one hidden neuron for each training pattern. There are no training parameters such as learning rate and momentum, but there is a smoothing factor, that is applied after the network is trained. For GRNN networks, the number of neurons in the hidden layer is usually the number of patterns in the training set because the hidden layer consists of one neuron for each pattern in the training set. You can make it larger if you may want to add more patterns, but don't make it smaller.

- GRNN Training Criteria

General Regression Neural Networks (GRNN) work by measuring how far a given sample pattern is from patterns in the training set in N dimensional space, where N is 35, the number of inputs in the problem.

When a new pattern is presented to the network, that input pattern is compared in N dimensional space to all of the patterns in the training set to determine how far in distance it is from those patterns. The output that is predicted by the network is a proportional

amount of all of the outputs in the training set. The proportion is based upon how far the new pattern is from the given patterns in the training set.

- GRNN Learning

Use this module to train GRNN networks. GRNN is essentially trained after one pass of the training patterns, and it is capable of functioning after only a few training patterns have been entered. (Obviously, GRNN training improves as more patterns are added.)

The network computes the mean (average) squared error between the actual and predicted values for all outputs over all patterns. The way it works is that, the network first computes the squared error for each output in a pattern, totals them and then computes the mean. When the mean over all patterns in the training set (in our case) reaches 0.2509134, then the minimum mean squared error found after long time of training.

The implemented network was saved.

4.3.3. Apply the GRNN Network

We used this module to process a data file through a trained neural network to produce the network's predictions for each pattern in the file. A file of outputs is produced. When we include actual values in the file, the module gives us check boxes to include actual values and the differences between the actual answers and the network's answers in the outputs file. The actual values and differences will be displayed for each output. The order of display is actual values, followed by predicted values, followed by differences.

When applying GRNN networks, we need to supply a smoothing factor required by the algorithm that affects the value of the output. For GRNN networks, the smoothing factor must be greater than 0 and can usually range from 0.01 to 1 with good results. We need to

experiment to determine which smoothing factor is most appropriate for our data. Fortunately, no retraining is required to change smoothing factors, because the value is specified when the network is applied. After experiment, it was found that, the appropriate value for our case is 0.01.

4.3.4. Output File

We used the Rules module to post-process the network's predictions. For example, if your outputs are categories and in the range zero to one, and you want any value in column 1 greater than .5 to be true (1), your rule might be:

If column 1 > = .5, then column 1 = 1,
else column 1 = 0.

In this way we can set our threshold for determining when the pattern is included in an output category. We used, in our problem, four rules, three of them are for the predicted topology (network 1), and one for the predicted cabling (network 2) as following:

If network (2) <15, then network (2)= 10,
else network (2)=20.

If network (1) < 1.5, then network (1)= 1.

If network (1) > = 1.5,
And network (1) < 2.5, then network (1)= 2.

If network (1) > = 2.5, then network (1)= 3.

Again, we can apply the Symbol Translate module on the output file after applying the rules above. At this phase of translation, we translate the numeric values to text, and the translation is done on the opposite way of that done in the input file (described in section 4.3.1).

Use the Examine Data module (also known as the Datagrid) to view files after network processing. The .OUT file is the default file that is passed to the Datagrid.

Chapter five: RESULS

Table 5.1. Datagrid input file

Measure	nodes number	nodes spacing	topology	cabling
100	2	0.10	3	10
200	2	0.50	3	10
400	2	1.00	3	20
800	2	2.00	3	20
1000	2	5.00	3	20
100	5	0.10	2	10
200	5	0.20	2	10
400	5	0.30	1	10
800	5	0.40	1	10
1000	5	0.50	1	20
100	8	0.10	2	10
200	8	0.45	2	10
400	8	1.20	2	20
800	8	3.00	1	20
1000	8	9.00	1	20
100	10	1.00	2	10
200	10	2.00	2	10
400	10	4.00	1	10
800	10	10.00	1	20
1000	10	19.00	1	20
100	12	5.00	2	20
200	12	11.00	1	20
400	12	21.00	1	20
800	12	48.00	1	20
1000	12	77.00	1	20
100	15	1.50	2	10

200	15	4.30	2	10
400	15	10.00	1	20
800	15	18.00	1	20
1000	15	33.00	1	20
100	20	0.50	1	10
200	20	0.90	1	10
400	20	2.10	2	20
800	20	8.00	1	20
1000	20	24.00	1	20

Table 5.1. Illustrates the input file fed into neural network after the text translated into numeric value. Historical patterns (records), that gives the network examples of correct classifications were chosen.

Measure taken from 100Mbs to 1000Mbs between switches. The number of nodes (switches) ranging from two nodes up to twenty nodes and assuming that the spacing between these node was between 0.1 Km to 77 km.

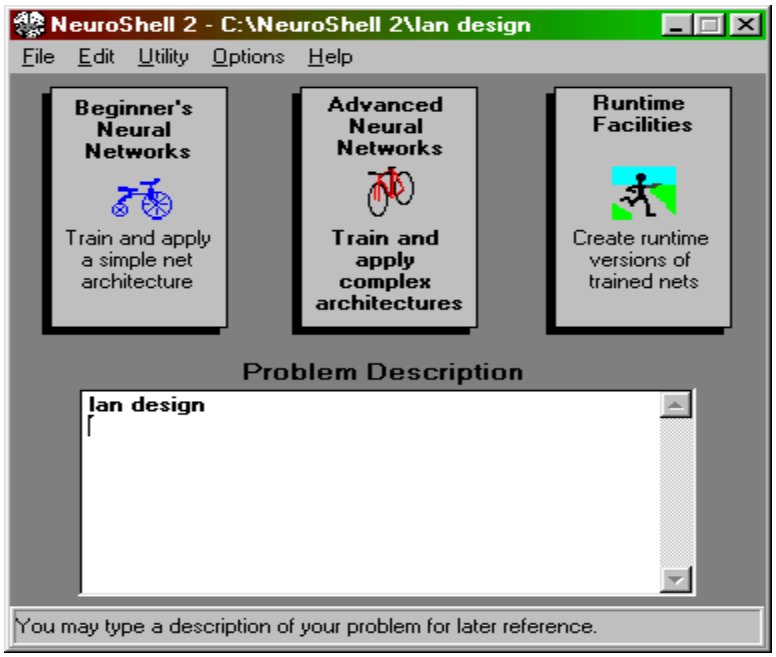


Figure 5.1. Initial module.

In fig. 5.1. the initial module is shown, in which the problem description can be defined.

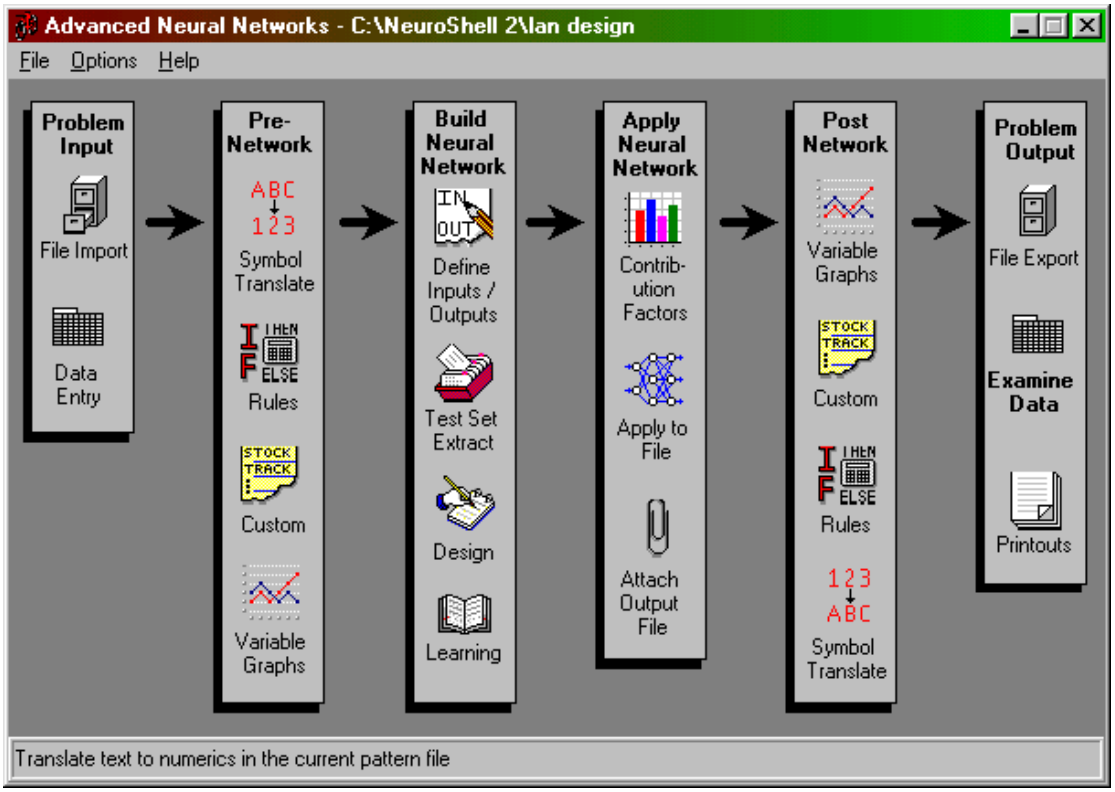


Figure 5.2. The Advanced Neural Networks Module

This module used for difficult problems, since somewhat better results may be got from those of initial module, our problem was considered to be difficult.

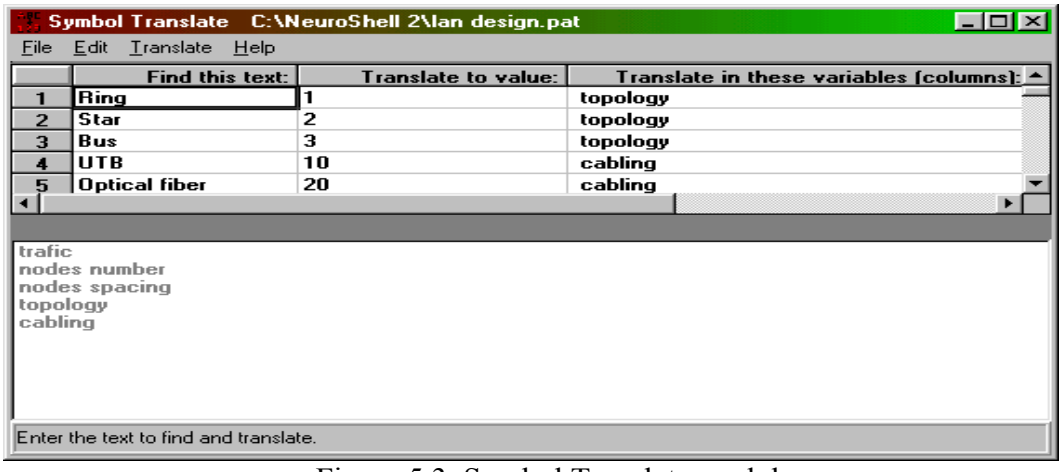
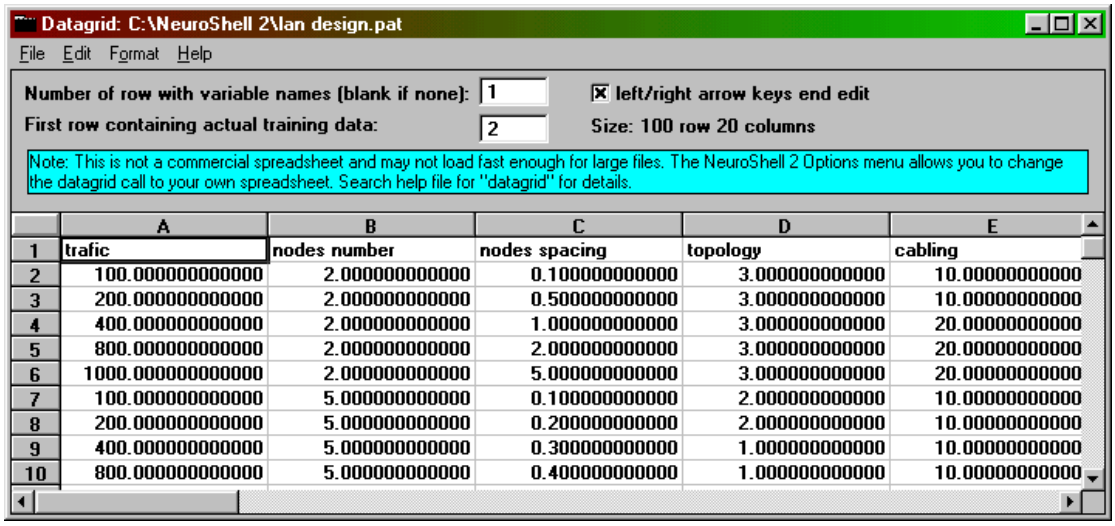


Figure 5.3. Symbol Translate module

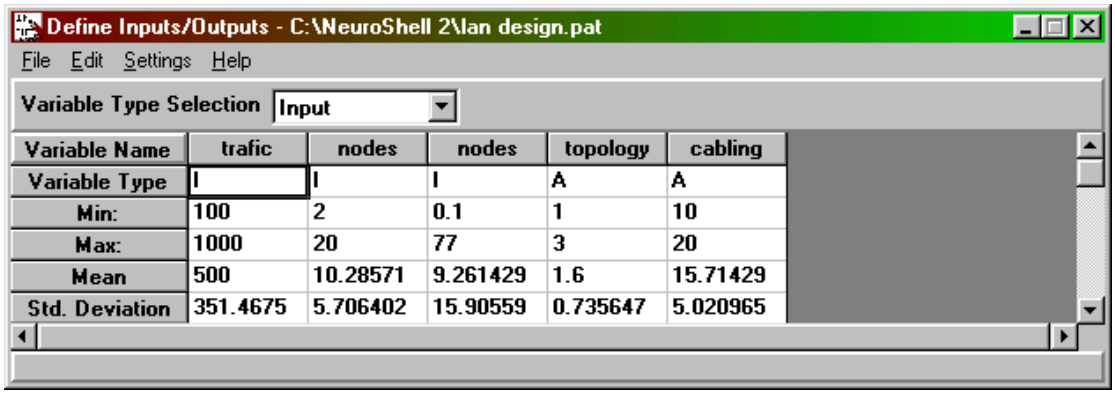
As discussed in chapter four-section 4.3.1, the symbol translation module shown in fig. 5.3 does the symbol translation.



	A	B	C	D	E
1	trafic	nodes number	nodes spacing	topology	cabling
2	100.000000000000	2.000000000000	0.100000000000	3.000000000000	10.000000000000
3	200.000000000000	2.000000000000	0.500000000000	3.000000000000	10.000000000000
4	400.000000000000	2.000000000000	1.000000000000	3.000000000000	20.000000000000
5	800.000000000000	2.000000000000	2.000000000000	3.000000000000	20.000000000000
6	1000.000000000000	2.000000000000	5.000000000000	3.000000000000	20.000000000000
7	100.000000000000	5.000000000000	0.100000000000	2.000000000000	10.000000000000
8	200.000000000000	5.000000000000	0.200000000000	2.000000000000	10.000000000000
9	400.000000000000	5.000000000000	0.300000000000	1.000000000000	10.000000000000
10	800.000000000000	5.000000000000	0.400000000000	1.000000000000	10.000000000000

Figure 5.4. Datagrid spreadsheet

Samples of data after implementing the symbol translation are presented in fig. 5.4. these data constitutes the pattern file, which is used to learn the neural network.



Variable Name	trafic	nodes	nodes	topology	cabling
Variable Type	I	I	I	A	A
Min:	100	2	0.1	1	10
Max:	1000	20	77	3	20
Mean	500	10.28571	9.261429	1.6	15.71429
Std. Deviation	351.4675	5.706402	15.90559	0.735647	5.020965

Figure 5.5. Define Inputs/Outputs module

In figure5.5 the first three variables in the pattern file (traffic, node number and nodes spacing) are defined as inputs, which are the variables used to make the prediction or classification. The last two variables (cabling and topology) are defined as actual output (the result), the network is trying to learn to predict.

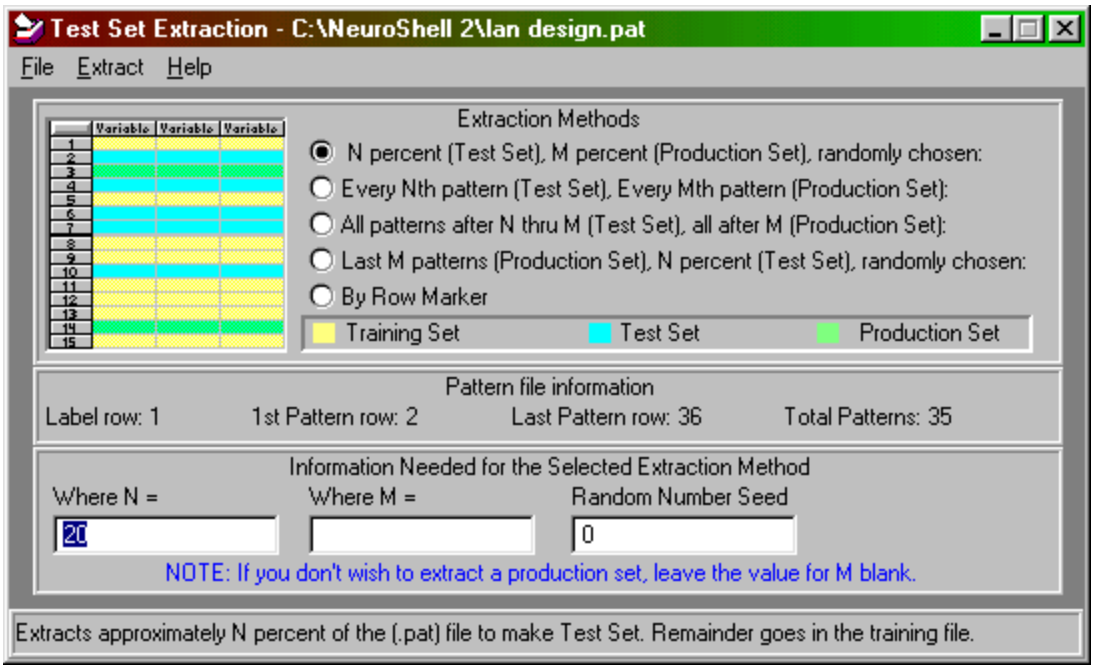


Figure 5.6. Test Set Extraction module

As shown in figure5.6, 20 percent of the pattern file was extracted as a test set. This extraction is useful to test the trained network if it can manipulate never seen data or not.

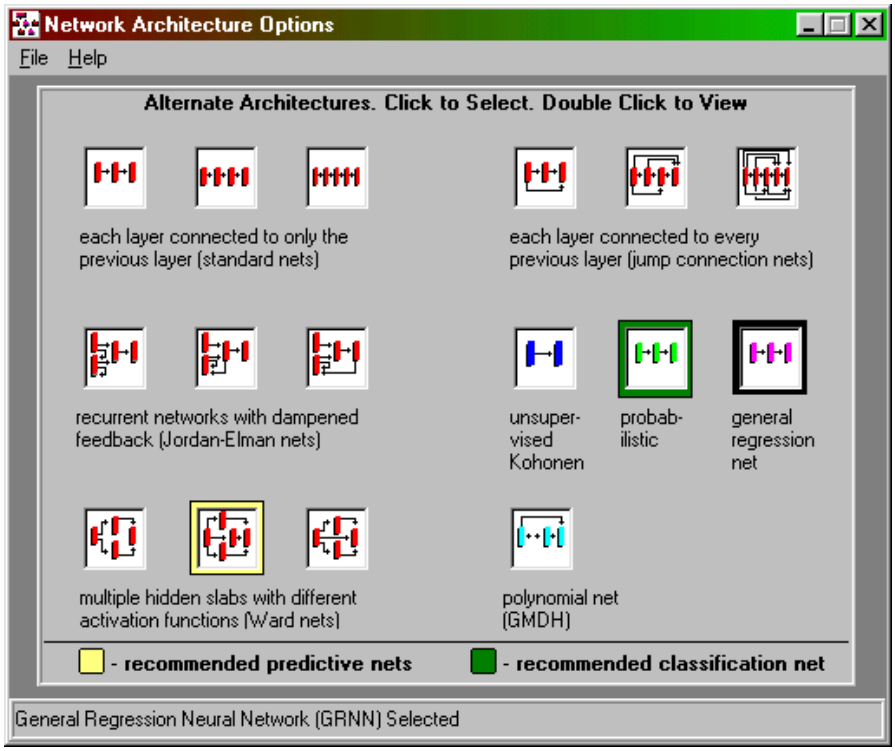


Figure 5.7. Network Architecture options

From the alternative architectures shown in figure 5.7, it was found that the GRNN network is more suitable for our problem than other networks.

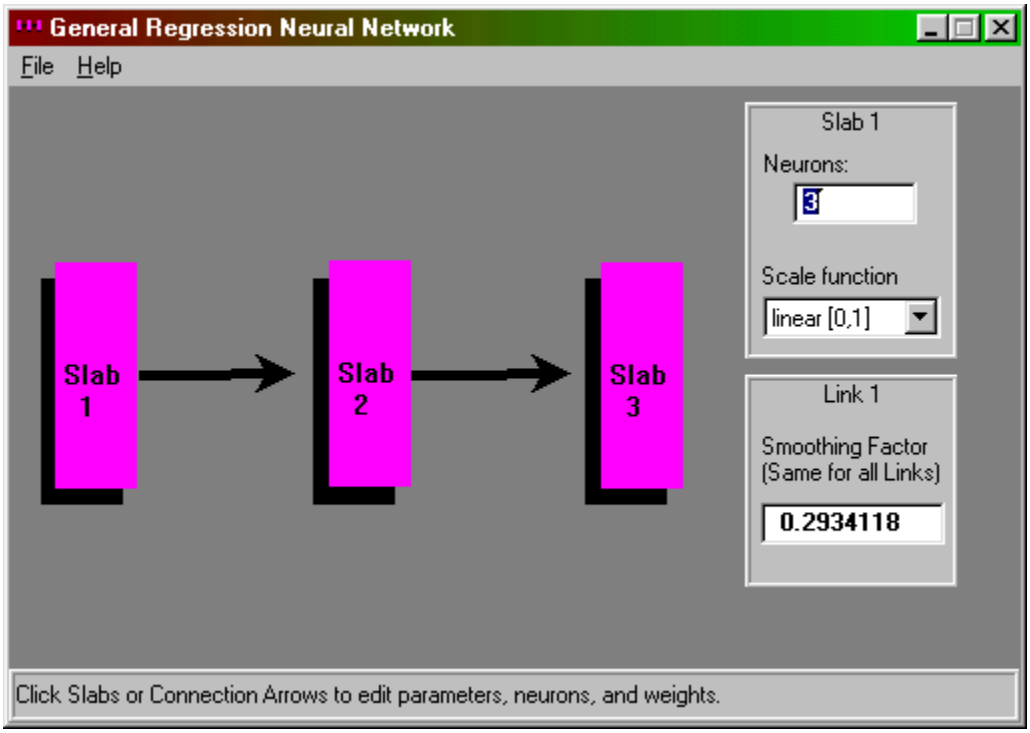


Figure 5.8. GRNN

We can change the default values of the neural network parameters such as the number of neurons on each slab (layer), the scale function and the smoothing factor, noting that the smoothing factor can be changed in the network processing phase.

The number of neurons in the input layer (Slab 1) is the number of inputs in our problem, and the number of neurons in the output layer (Slab 3) corresponds to the number of outputs. The smoothing factor is a parameter of GRNN networks. Higher smoothing factors cause more relaxed surface fits through the data.

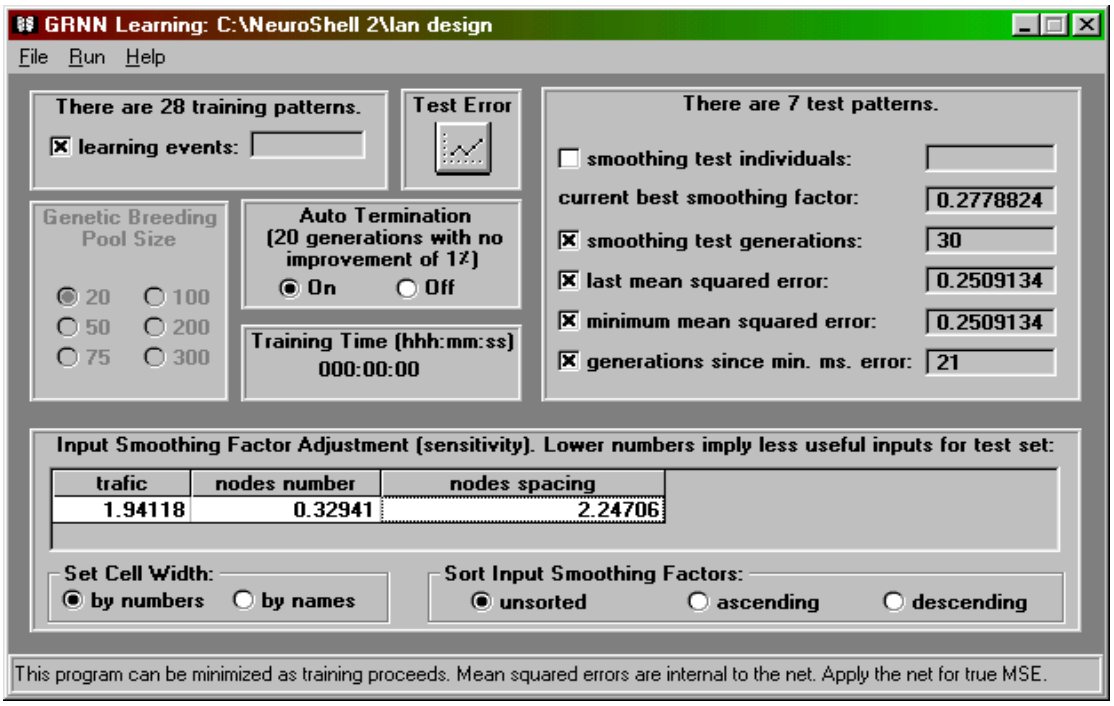


Figure 5.9. GRNN learning module

The mean squared error is the standard statistical technique for determining closeness of fit.) Calibration computes the squared error for each output in a pattern, totals them and then computes the mean of that number over all patterns in the test set Fig. 5.9 shows that the minimum mean squared error is 0.2509134 when the smoothing factor is 0.2778824.

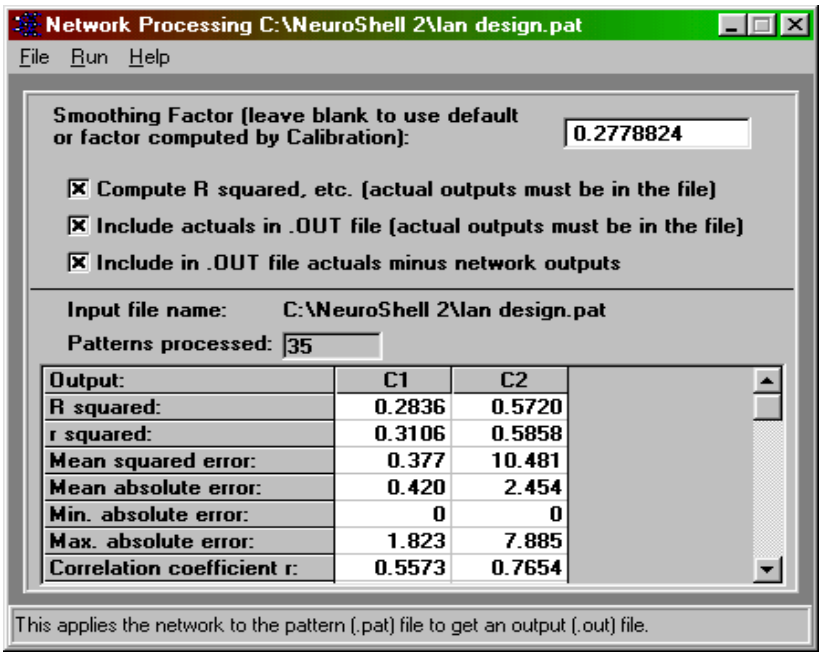


Figure 5.10. (a) Network Processing module

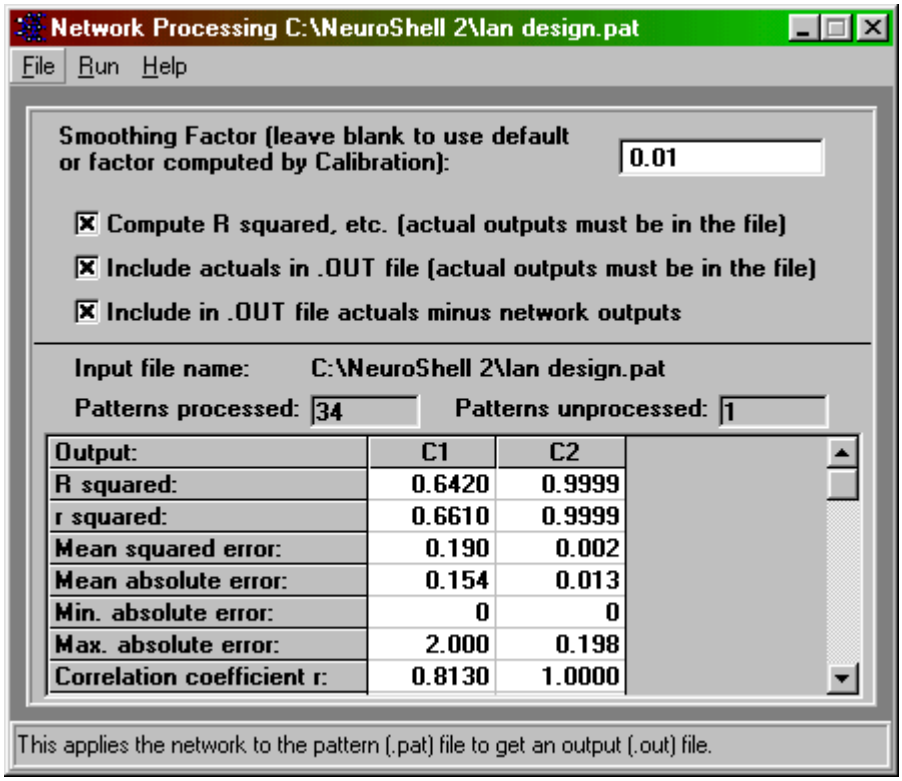
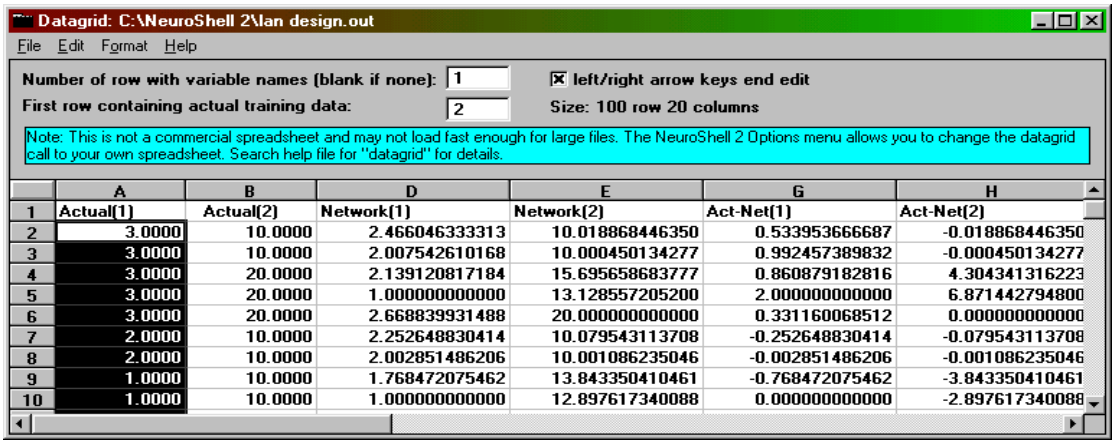


Figure 5.10. (b) Network Processing module

Figures 5.10. (a) and 5.10. (b) illustrate the effect of the smoothing factor on R squared values. In the first figure we used the default smoothing factor 0.2778824, which gives R squared=0.2836 for the first output and 0.5720 for the second output. In the second figure, after multiple experiments, we prefer to choose 0.01 as smoothing factor, where the R squared values for the first output is increased to 0.6420 and for the second output is increased to 0.9999, which means better generalization.



	A	B	D	E	G	H
1	Actual(1)	Actual(2)	Network(1)	Network(2)	Act-Net(1)	Act-Net(2)
2	3.0000	10.0000	2.466046333313	10.018868446350	0.533953666687	-0.018868446350
3	3.0000	10.0000	2.007542610168	10.000450134277	0.992457389832	-0.000450134277
4	3.0000	20.0000	2.139120817184	15.695658683777	0.860879182816	4.304341316223
5	3.0000	20.0000	1.000000000000	13.128557205200	2.000000000000	6.871442794800
6	3.0000	20.0000	2.668839931488	20.000000000000	0.331160068512	0.000000000000
7	2.0000	10.0000	2.252648830414	10.079543113708	-0.252648830414	-0.079543113708
8	2.0000	10.0000	2.002851486206	10.001086235046	-0.002851486206	-0.001086235046
9	1.0000	10.0000	1.768472075462	13.843350410461	-0.768472075462	-3.843350410461
10	1.0000	10.0000	1.000000000000	12.897617340088	0.000000000000	-2.897617340088

Figure 5.12. Datagrid (output file)

Samples of output file are illustrated in fig.5.12. The first two columns shows the actual outputs as entered in the input datagrid before data processed to the neural network. The followed two columns (network (1) and network (2)) contain the output prediction of the neural network (GRNN). The last two columns are the difference between the actual outputs and the outputs predicted by the network for all patterns.

Chapter Six: Conclusion

In this project the implementation of Neural Networks as a solution for LAN design, and the different phases of work to create a reliable system that would help us to decide the cabling and network topology to be used in our LAN have been discussed. The design for the backbone have been focused on, which is an important layer of the LAN.

As mentioned earlier, it is vital to collect the data from many and noticeably reliable sources, for it is going to be the learning data in the view point of Neural Networks, this data is used to train the software to the point that the error is minimum.

To achieve the best results, several Neural Network architectures were examined, and it was found that the General Regression Neural Networks (GRNN) model was the most suitable architecture for the application in question, where it generated lower values of error.

We used bandwidth, spacing between switches, and the number of nodes as inputs in our implementation that was discussed through this research , where as topology and cabling type were the outputs . However with the flexibility of Neural Networks it is recommended to introduce more inputs such as Maximum Transmission Unit (MTU), and delay, this would raise the efficiency of the application.

It is essential to mention that certain factors should still be considered by the designer in order to have an optimum LAN, these are such as the probability of network expansion, the type of switches used and their capacity, the CPU and memory capability of the work stations, and the building architecture where this LAN would reside.

Here we conclude that careful use of Neural Networks can provide a viable solution for LAN design , and with more factors considered it can help us in more decisions of our network , such as routing protocols to be used ..etc..

Reference:

1. Madron, t. Local Area Networks: New Technologies, Emerging standards. New York: Wiley, 1994.
2. Martin, J., Chapman, K., and Leben, J. Local Area Networks: Architectures and implementation. Englewood Cliffs, NJ: Prentice Hall, 1994.
3. McElroy, M. The Corporate Cabling Guide. Boston: Artech House, 1993.
4. Stallings, W. Local and Metropolitan Area Networks: Prentice Hall. New Jersey, 1997.
5. Stephens, G., and DedeK, J. Fiber Channel Menlo Park, CA: Ancot Corporation, 1995.
6. Martin, T., Howard, B., and Mark Beale. Neural Network Design. University of Colorado, 1996.